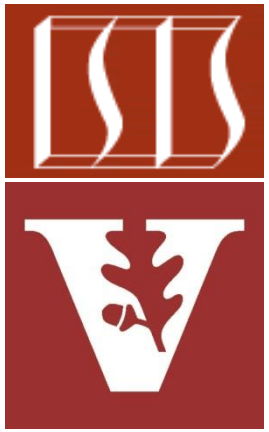
# The AsyncTask Framework: Introduction

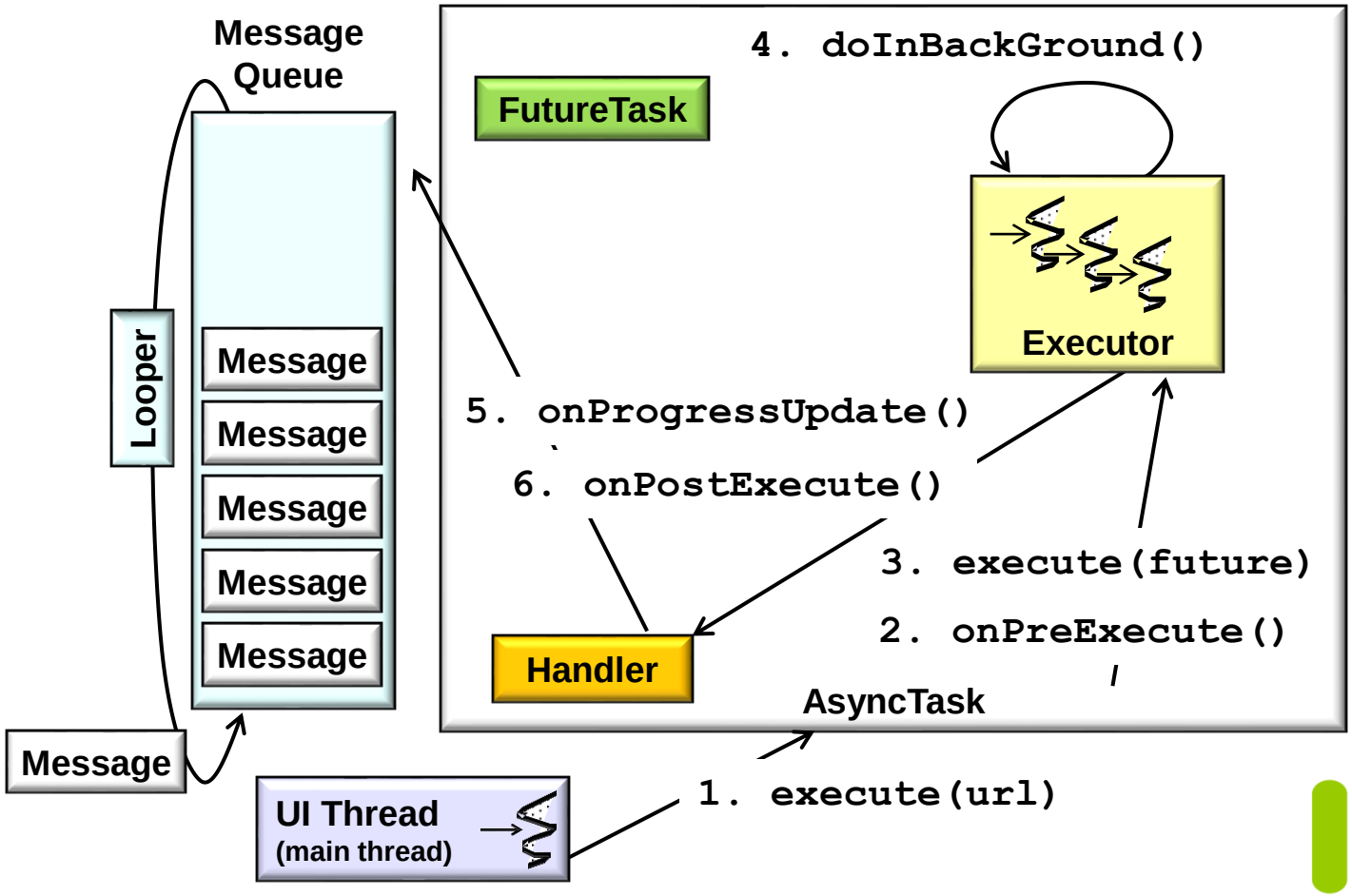


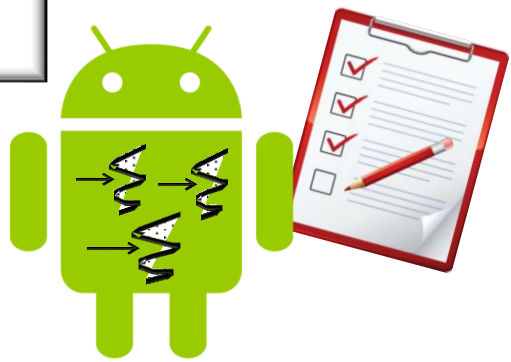

Douglas C. Schmidt
d.schmidt@vanderbilt.edu
www.dre.vanderbilt.edu/~schmidt

Institute for Software
Integrated Systems
Vanderbilt University
Nashville, Tennessee, USA


# Learning Objectives in this Part of the Lesson

- Recognize the capabilities provided by the Android AsyncTask framework



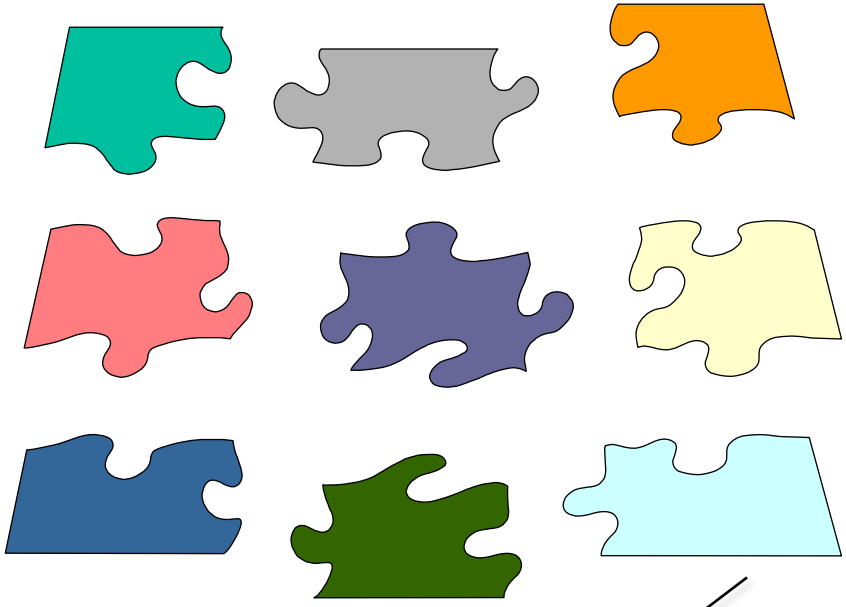



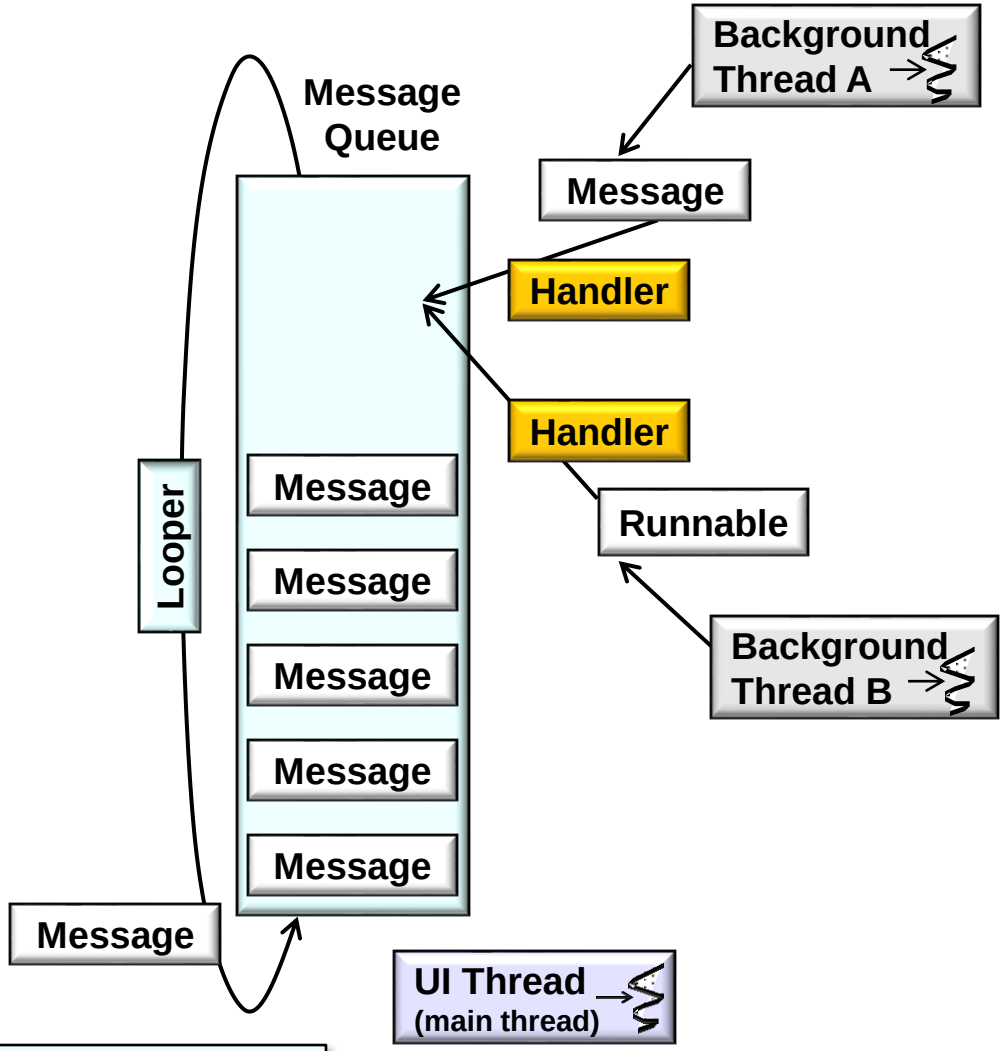
Allows apps to perform background operations & publish results on UI thread *without* manipulating threads, handlers, messages, or runnables

# Overview of the AsyncTask Framework

# Overview of the AsyncTask Framework

- Classes in HaMeR framework are loosely connected

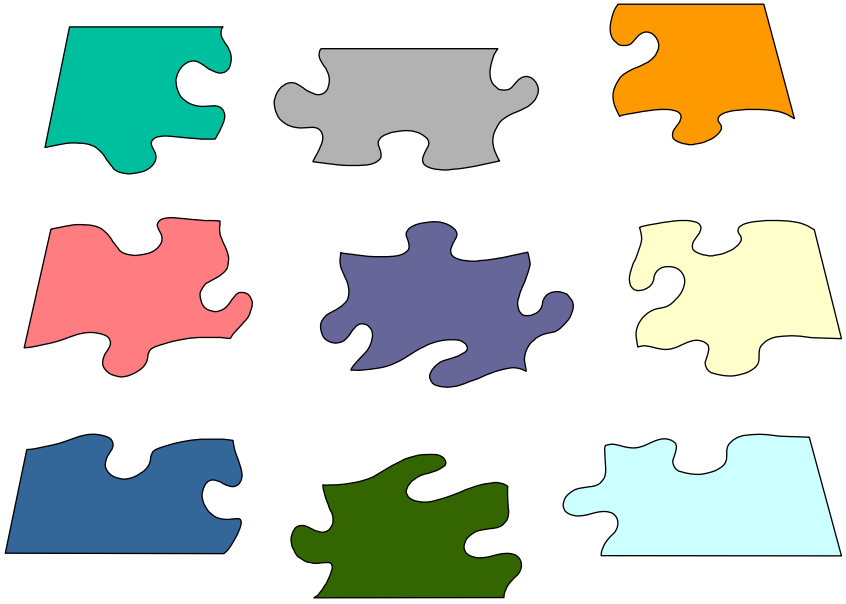


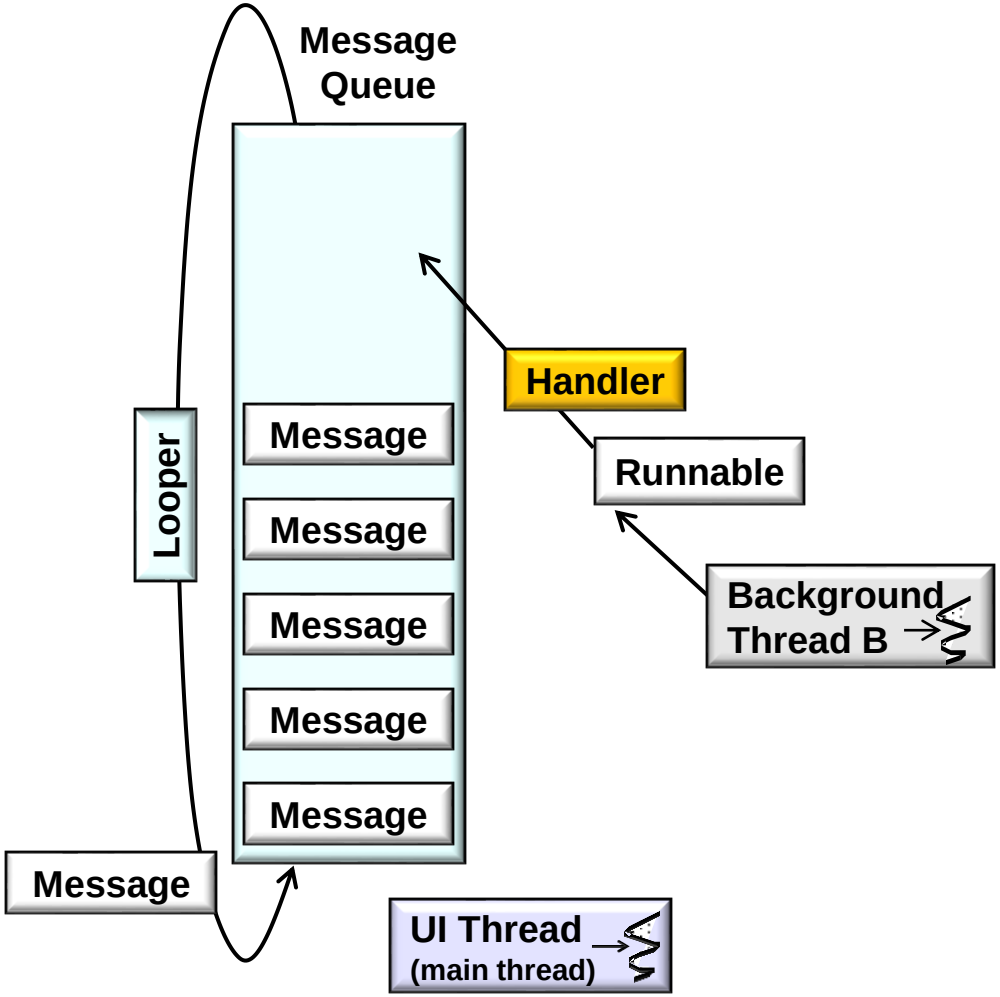



*e.g., it's not clear from the Android documentation that classes in the HaMeR framework are related*

# Overview of the AsyncTask Framework

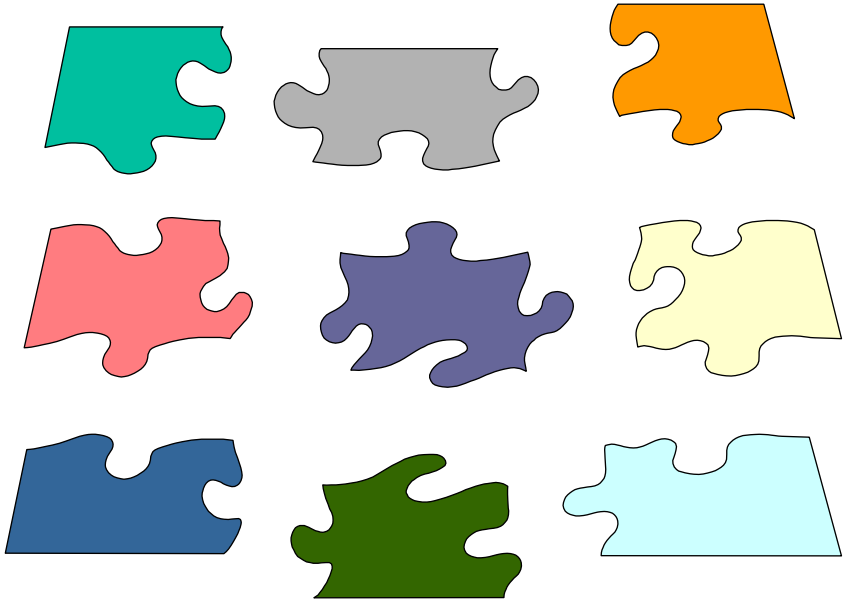
- Classes in HaMeR framework are loosely connected
  - This flexibility works well for simple concurrency use cases

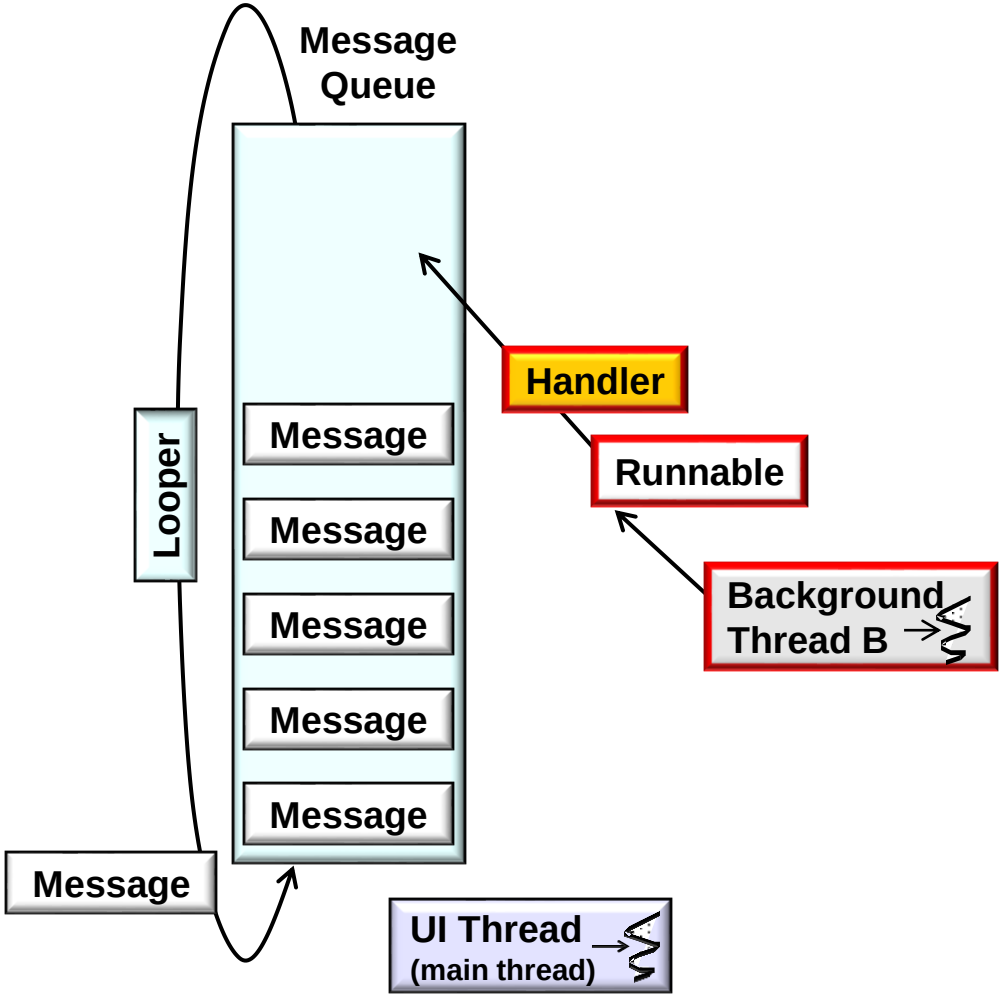




# Overview of the AsyncTask Framework

- Classes in HaMeR framework are loosely connected
  - This flexibility works well for simple concurrency use cases

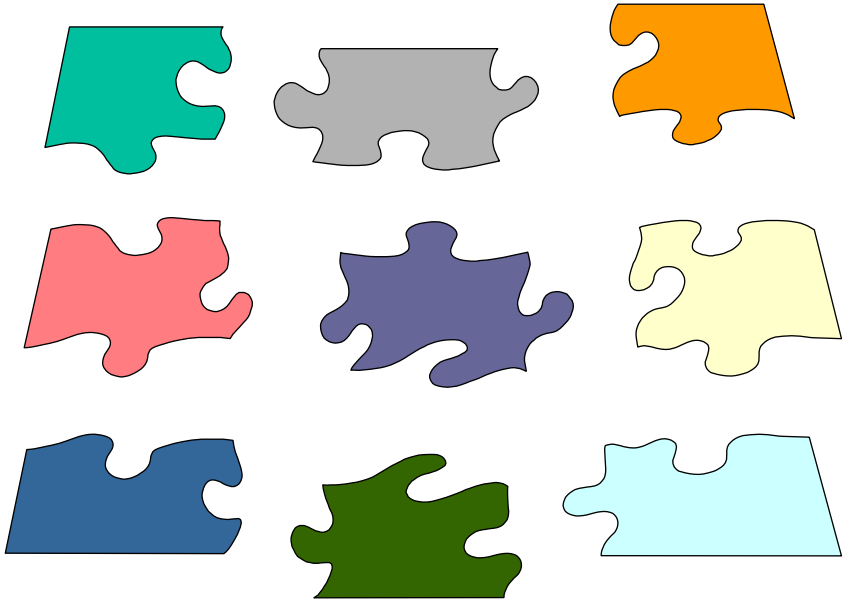


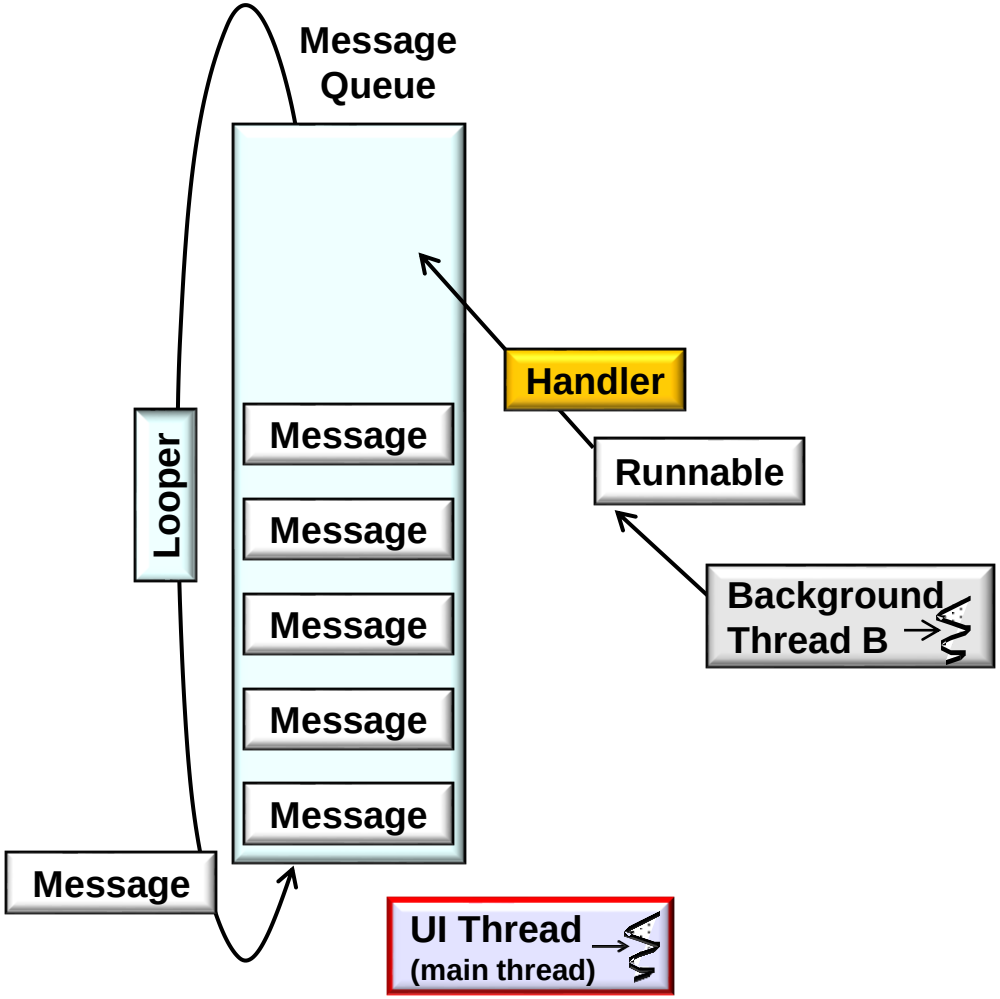



e.g., where a background thread posts a runnable to the UI thread…

# Overview of the AsyncTask Framework

- Classes in HaMeR framework are loosely connected
  - This flexibility works well for simple concurrency use cases

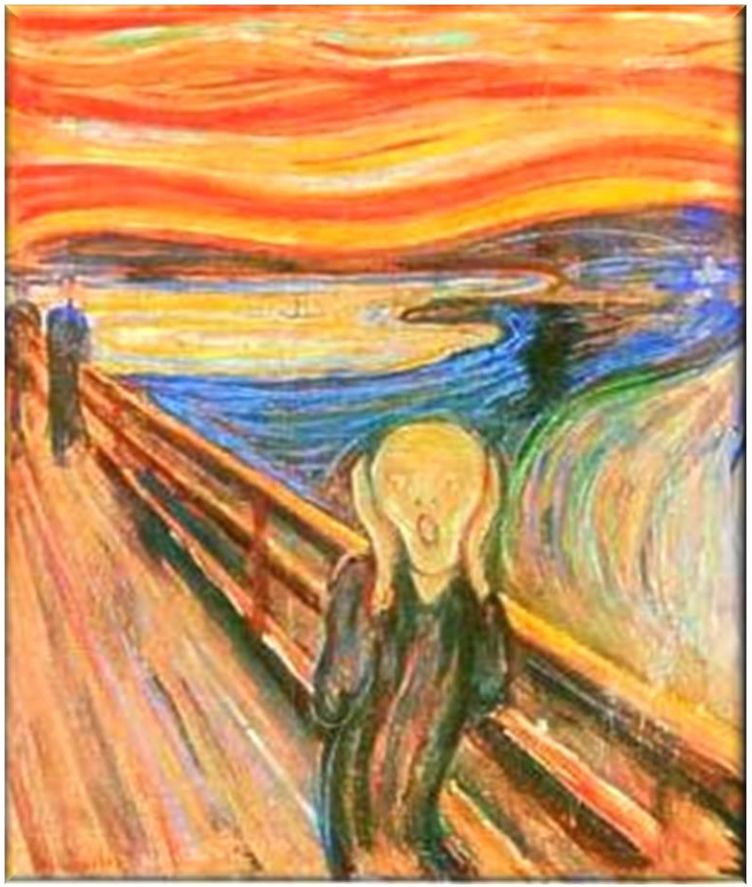


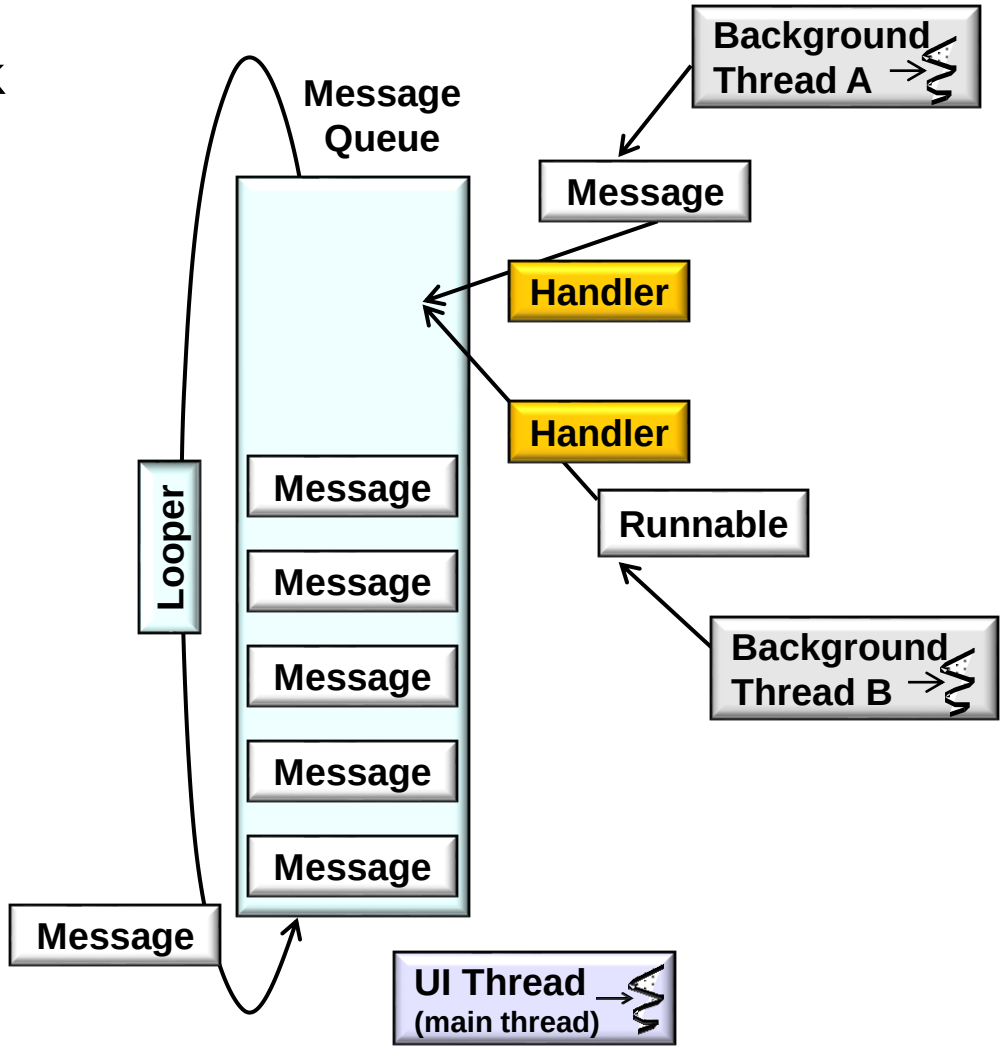



… & the UI thread dispatches the run() hook method of the runnable

# Overview of the AsyncTask Framework

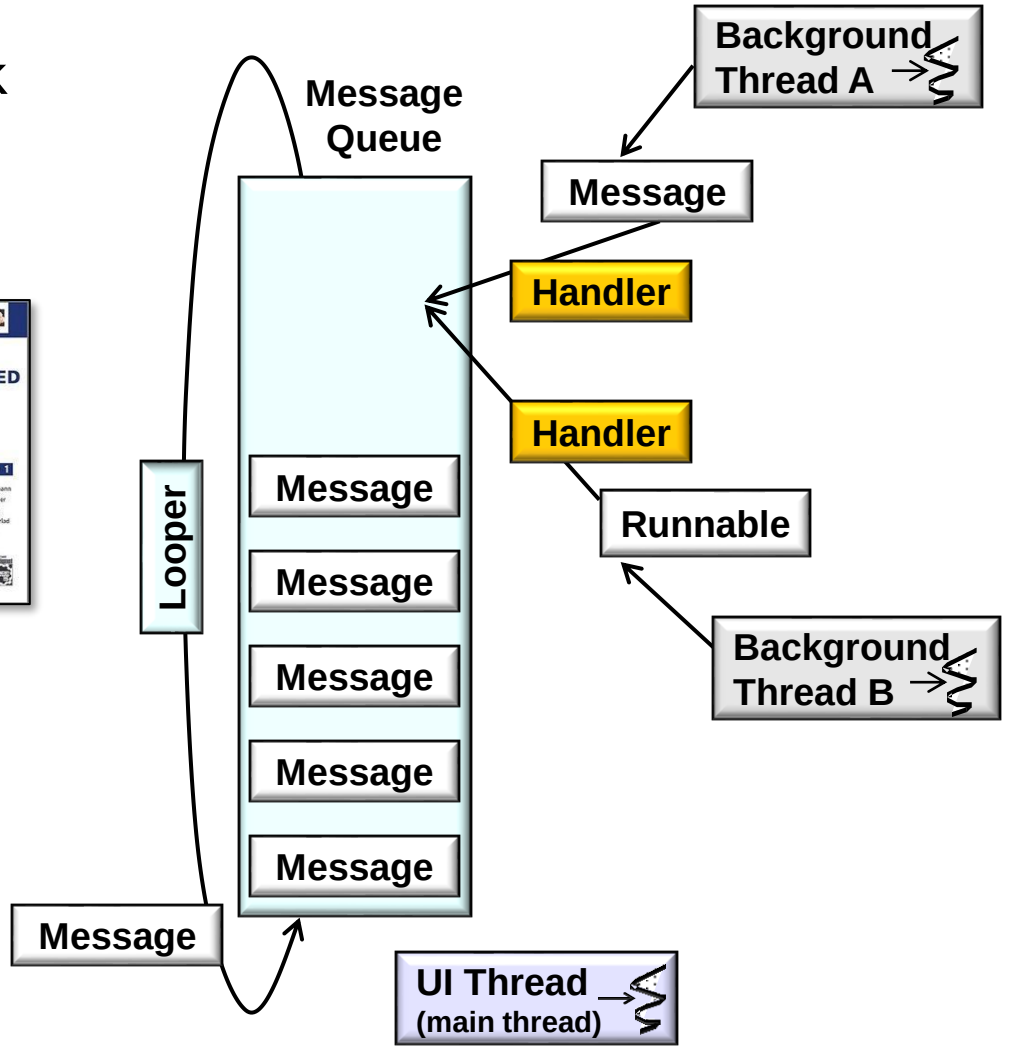
- However, there are drawbacks to the HaMeR concurrency framework





# Overview of the AsyncTask Framework

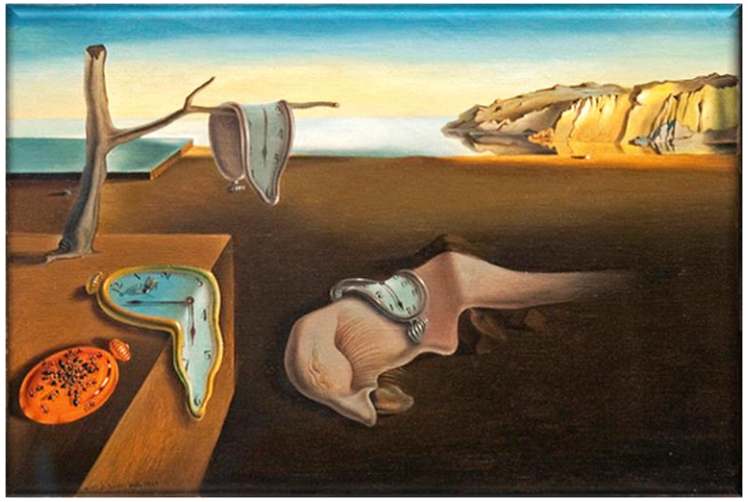
- However, there are drawbacks to the HaMeR concurrency framework
  - Must understand patterns to use this framework effectively

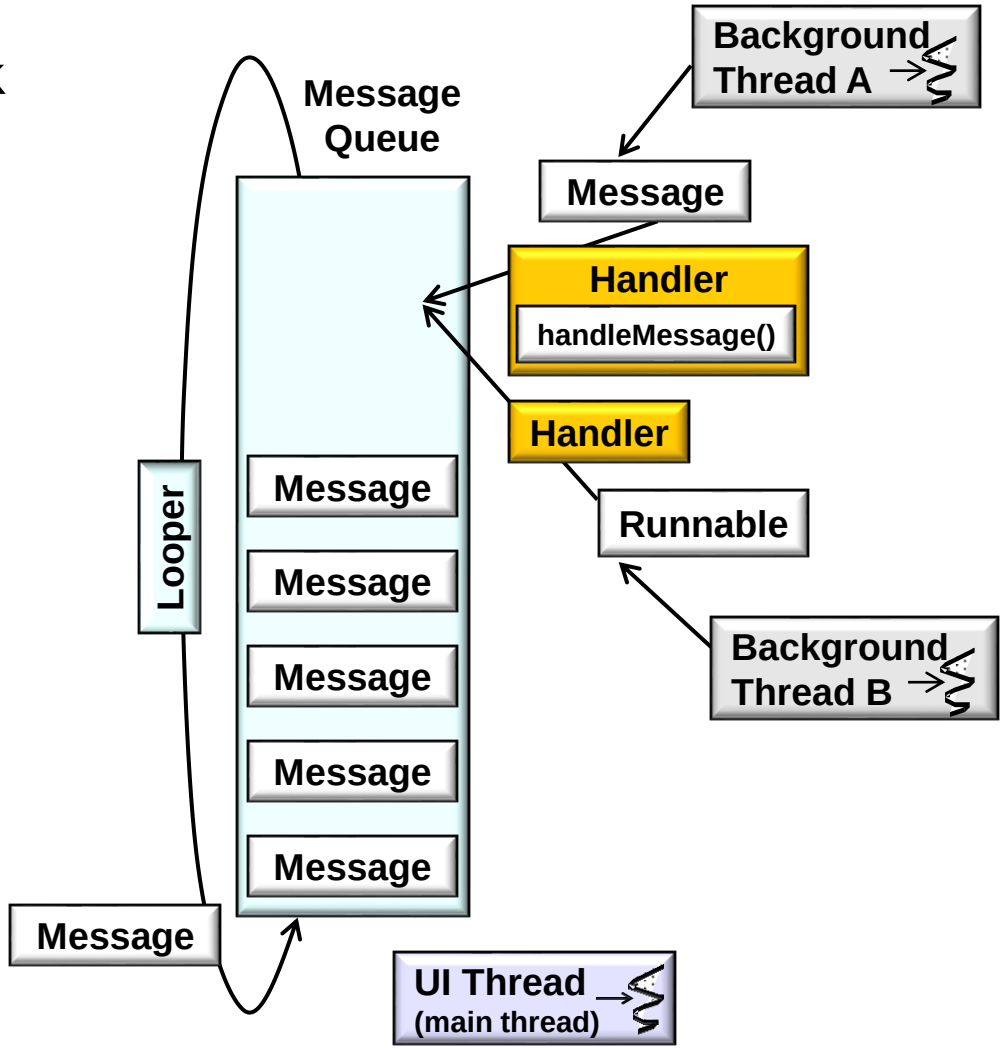




See en.wikipedia.org/wiki/Active_object & www.dre.vanderbilt.edu/~schmidt/CommandProcessor.pdf

# Overview of the AsyncTask Framework

- However, there are drawbacks to the HaMeR concurrency framework
  - Must understand patterns to use this framework effectively
  - Tedious & error-prone to use

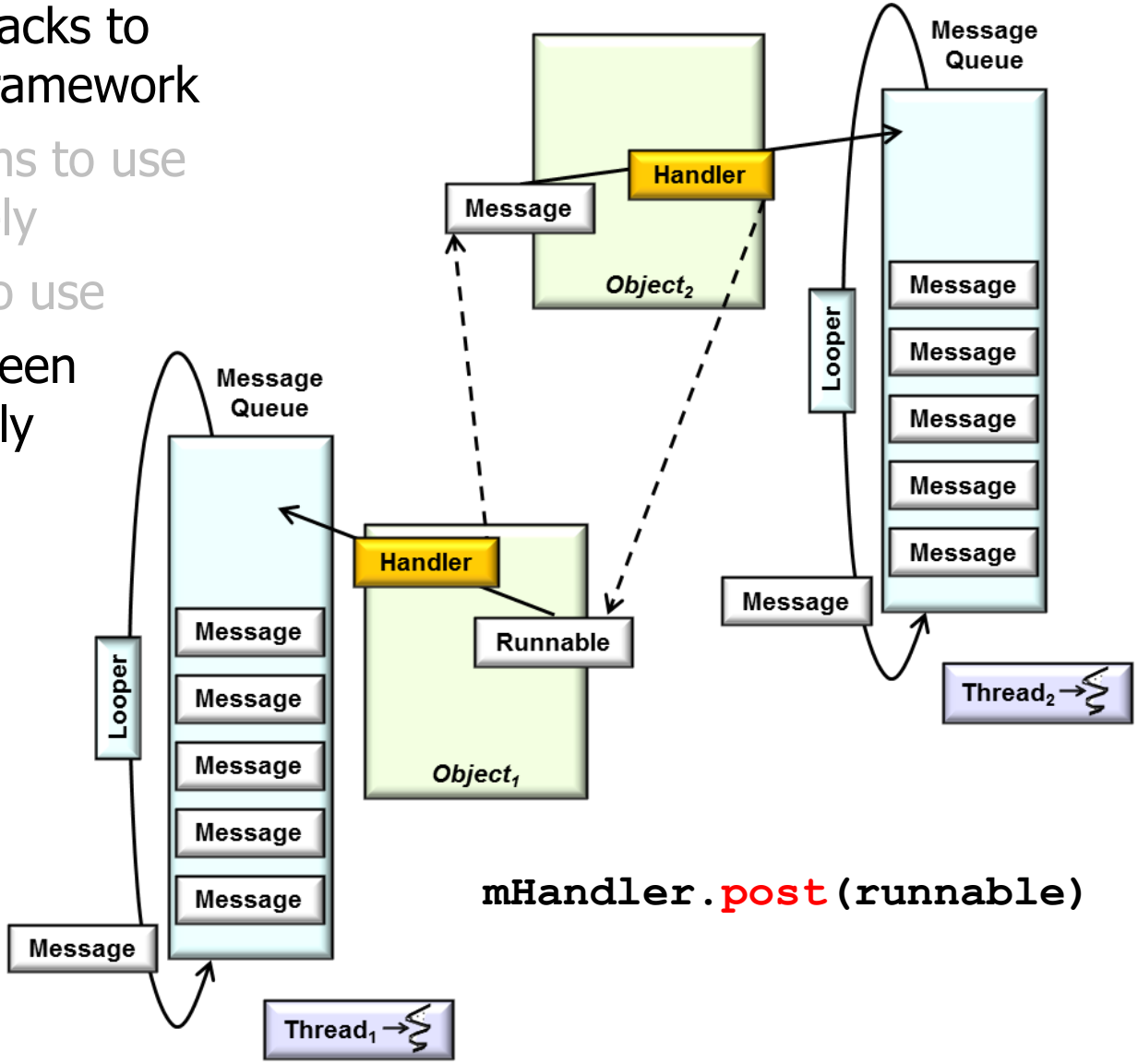





e.g., apps must understand how to manage the lifecycle of messages

# Overview of the AsyncTask Framework

- However, there are drawbacks to the HaMeR concurrency framework
  - Must understand patterns to use this framework effectively
  - Tedious & error-prone to use
  - All communication between threads must be explicitly programmed

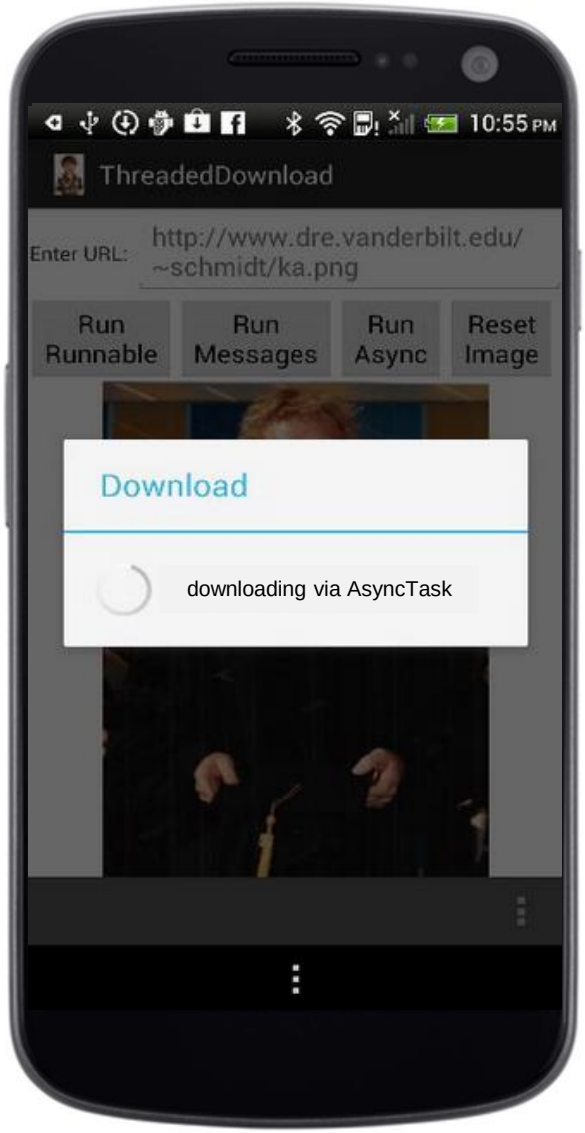



```
mHandler.sendMessage
        (message)
```

```
mHandler.post(runnable)
```

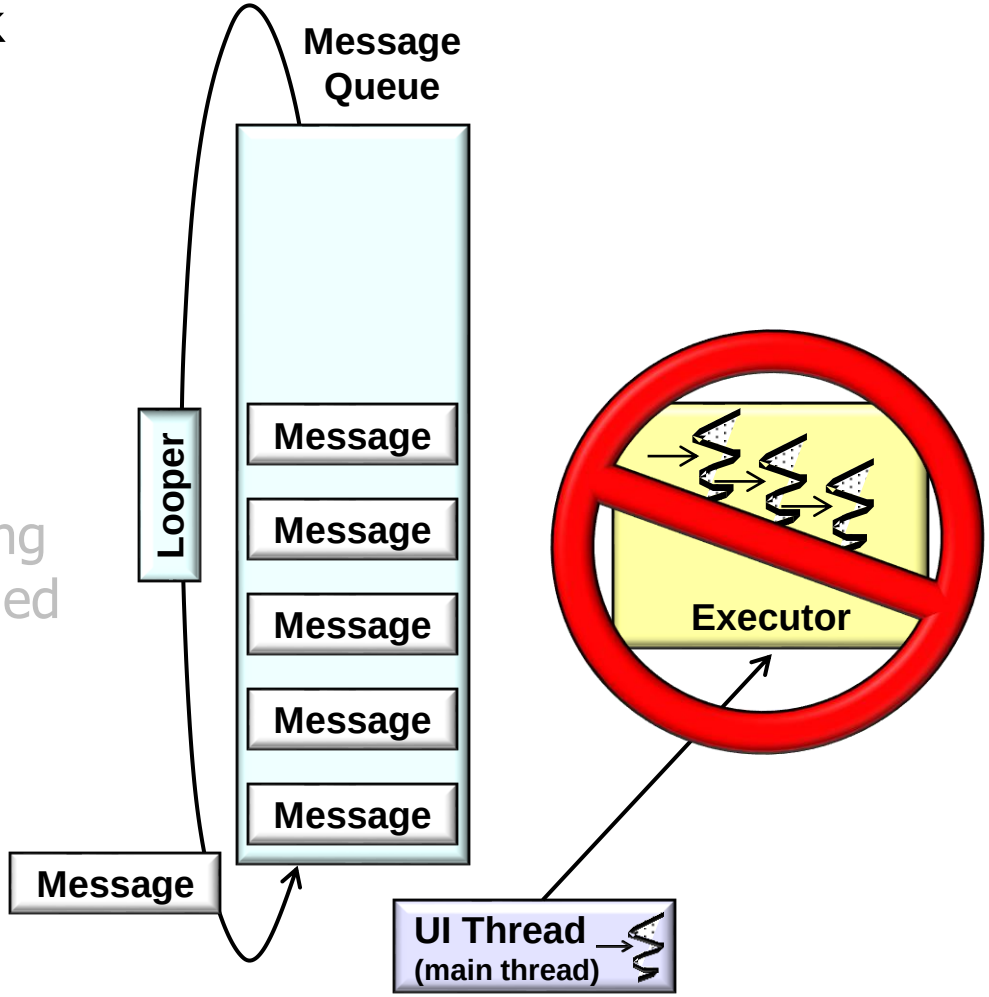
# Overview of the AsyncTask Framework

- However, there are drawbacks to the HaMeR concurrency framework
  - Must understand patterns to use this framework effectively
  - Tedious & error-prone to use
  - All communication between threads must be explicitly programmed
  - Any "pre" and/or "post" processing must also be explicitly programmed
    - e.g., starting & stopping a progress dialog box



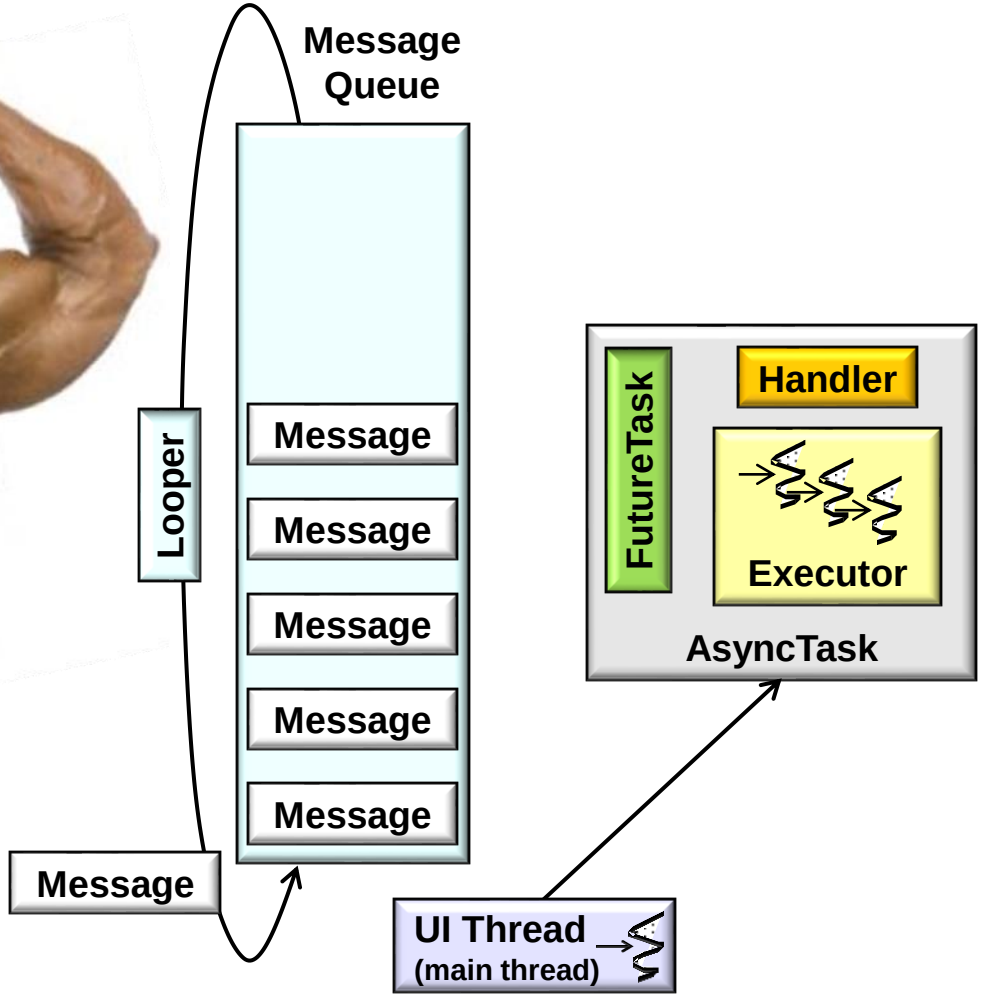
# Overview of the AsyncTask Framework

- However, there are drawbacks to the HaMeR concurrency framework
  - Must understand patterns to use this framework effectively
  - Tedious & error-prone to use
  - All communication between threads must be explicitly programmed
  - Any "pre" and/or "post" processing must also be explicitly programmed
  - Performance can't be scaled up transparently



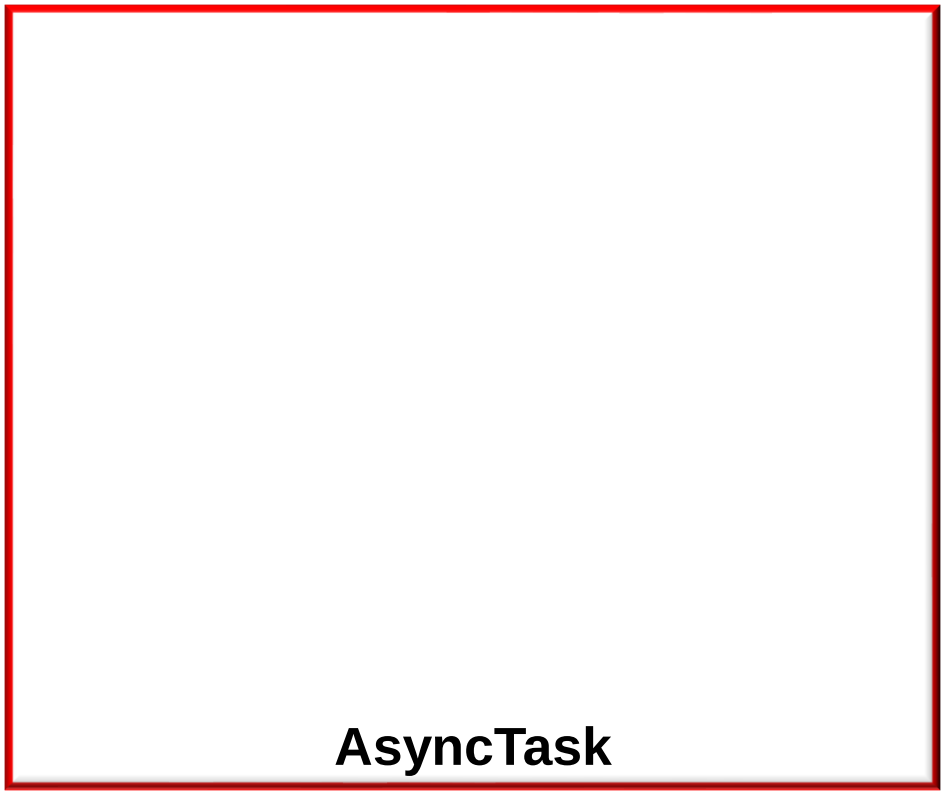
# Overview of the AsyncTask Framework

- In contrast, AsyncTask framework classes are more strongly connected



# Overview of the AsyncTask Framework

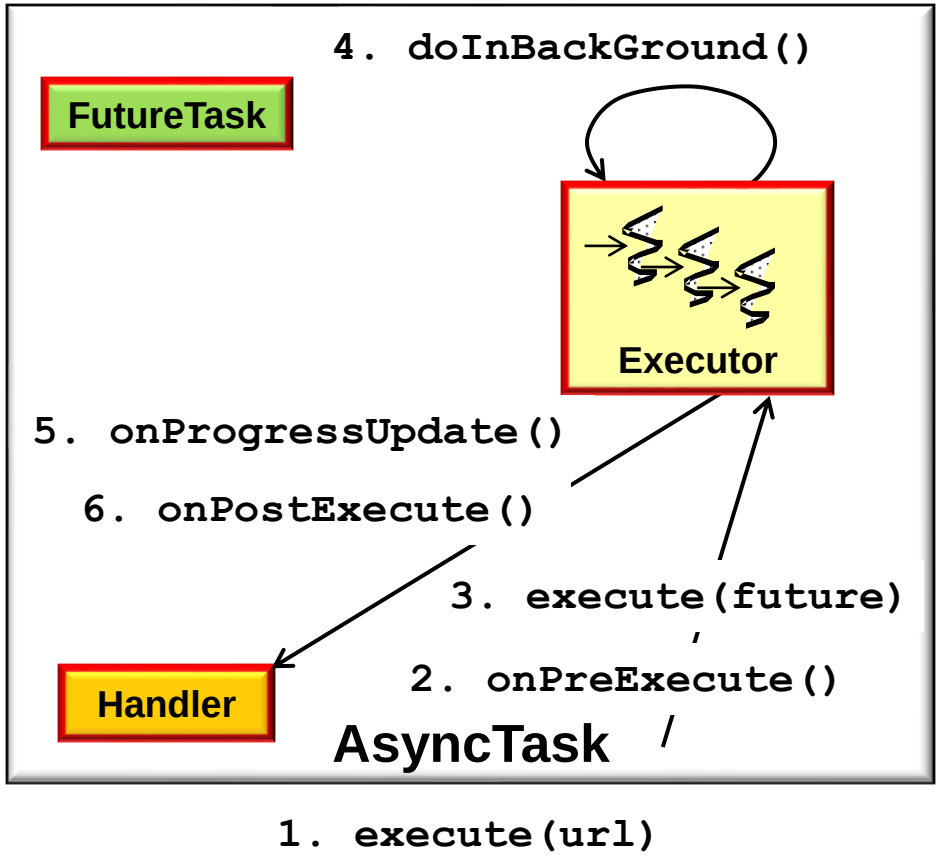
- In contrast, AsyncTask framework classes are more strongly connected
  - Complex framework details hidden via *Façade* pattern




See en.wikipedia.org/wiki/Facade_pattern

# Overview of the AsyncTask Framework

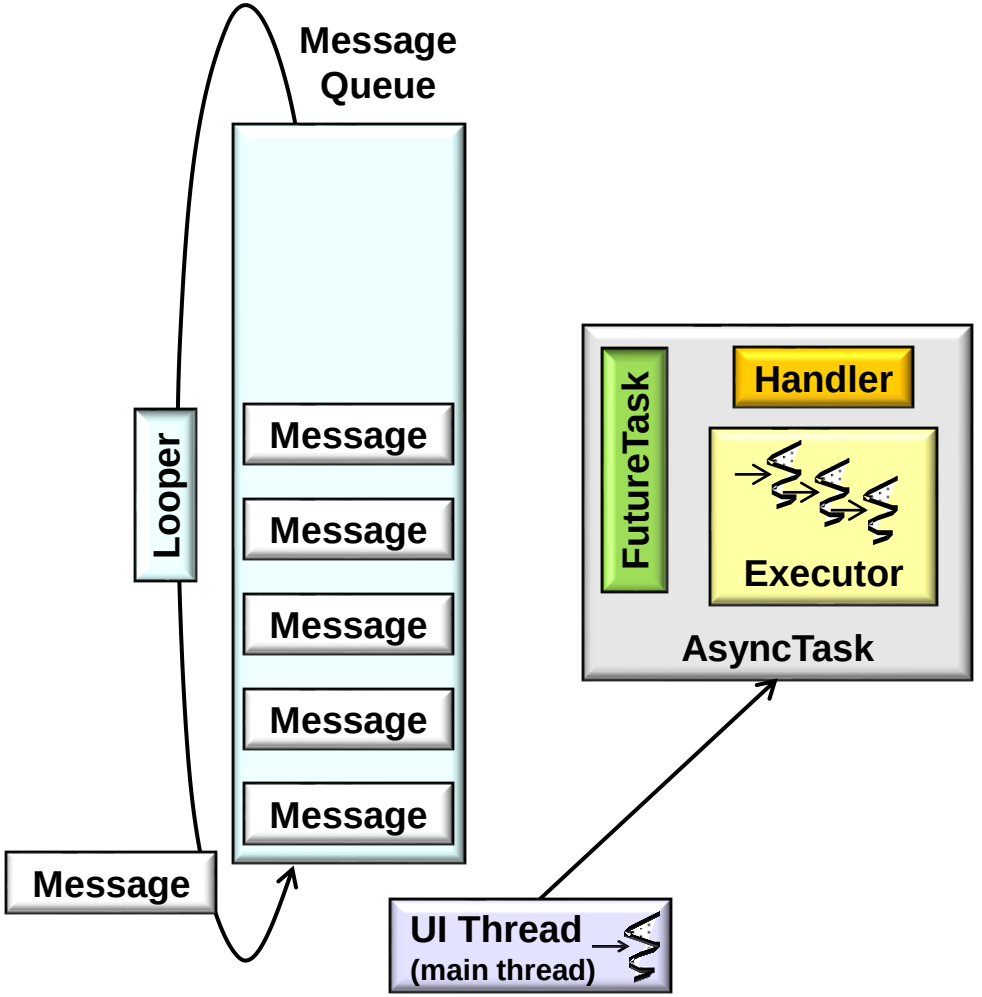
- In contrast, AsyncTask framework classes are more strongly connected
  - Complex framework details hidden via *Façade* pattern
    - Encapsulates the complicated classes in AsyncTask framework with a simpler interface




See developer.android.com/reference/android/os/AsyncTask.html

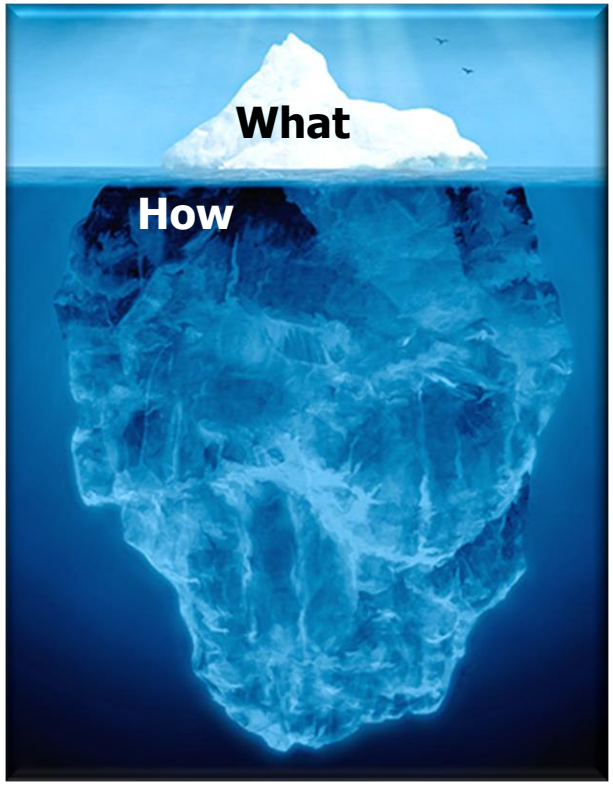
# Overview of the AsyncTask Framework

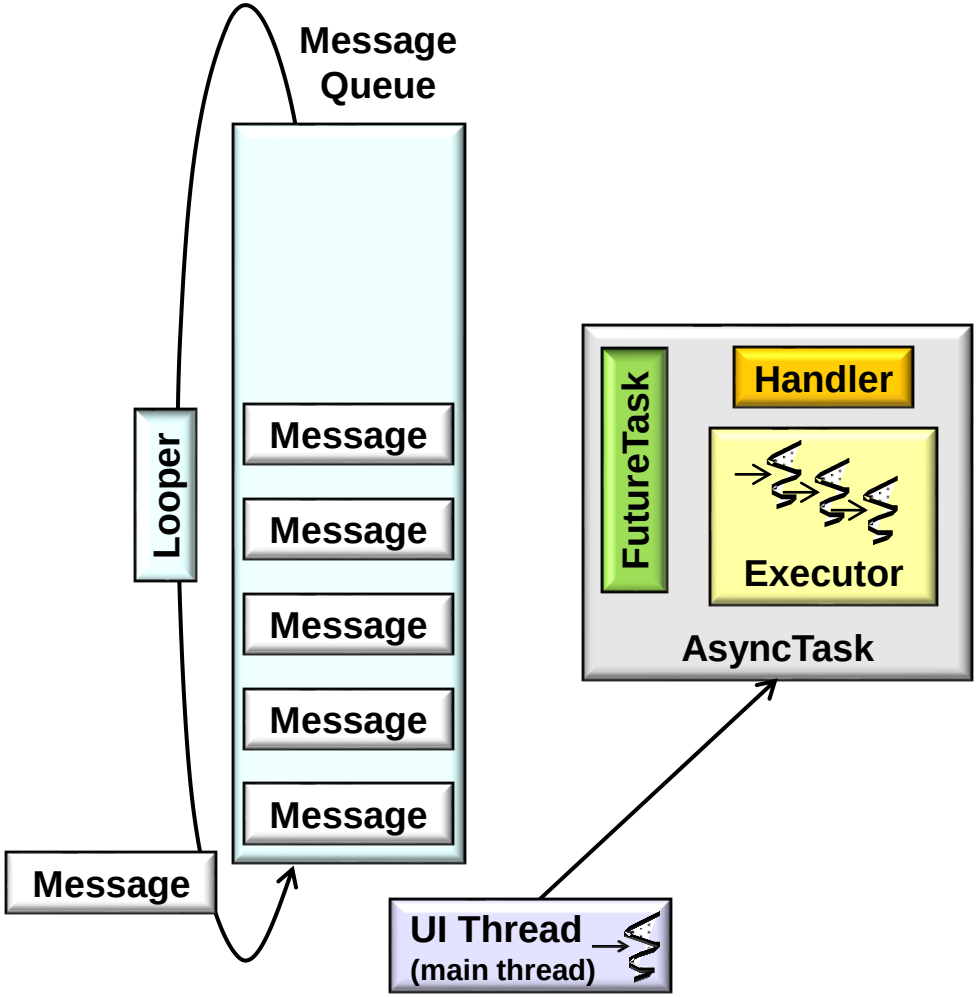
- In contrast, AsyncTask framework classes are more strongly connected
  - Complex framework details hidden via *Façade* pattern
  - Yields a smaller "surface area"



# Overview of the AsyncTask Framework

- In contrast, AsyncTask framework classes are more strongly connected
  - Complex framework details hidden via *Façade* pattern
  - Yields a smaller "surface area"

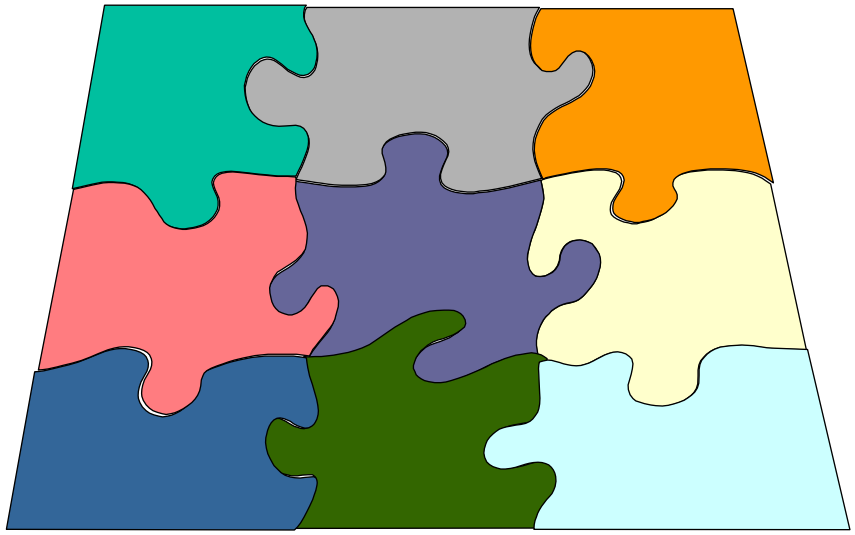


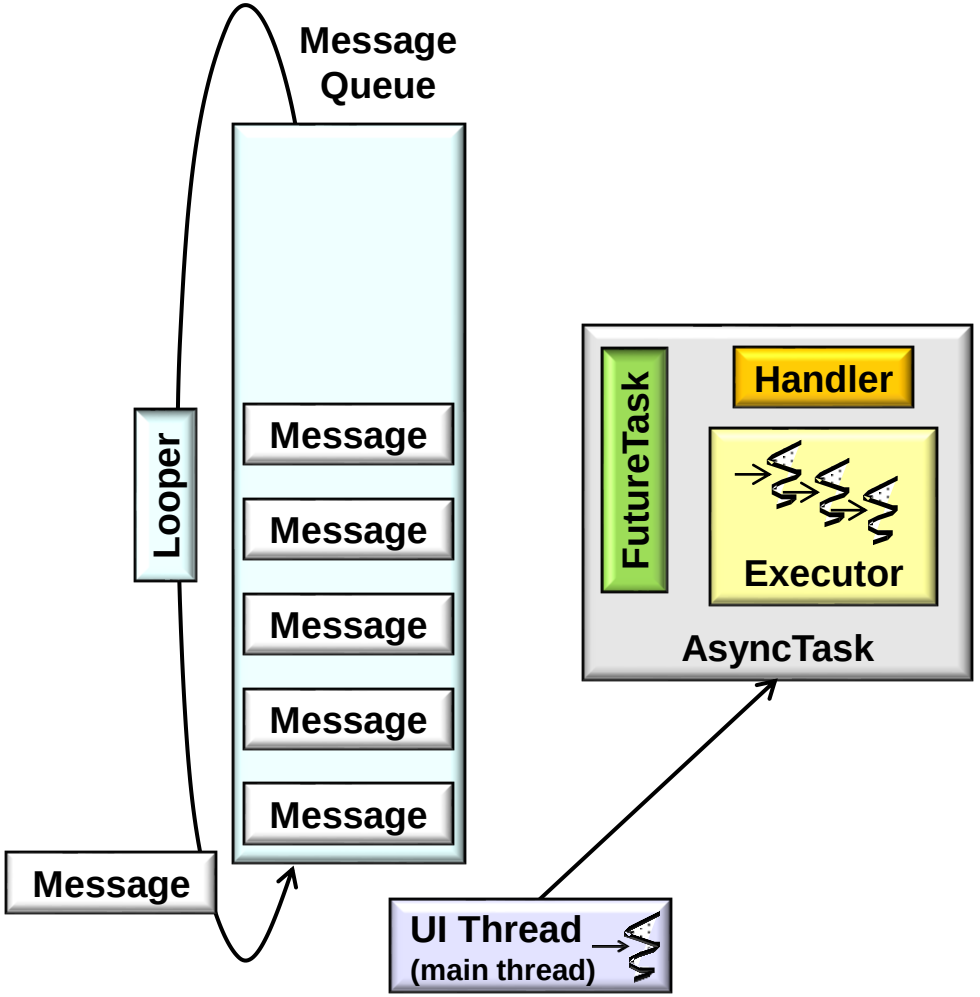




i.e., programmers can focus on the "what" not the "how"

# Overview of the AsyncTask Framework

- In contrast, AsyncTask framework classes are more strongly connected
  - Complex framework details hidden via *Façade* pattern
  - Yields a smaller "surface area"
  - Tasks run concurrently, *without* manipulating threads, handlers, messages, or runnables directly

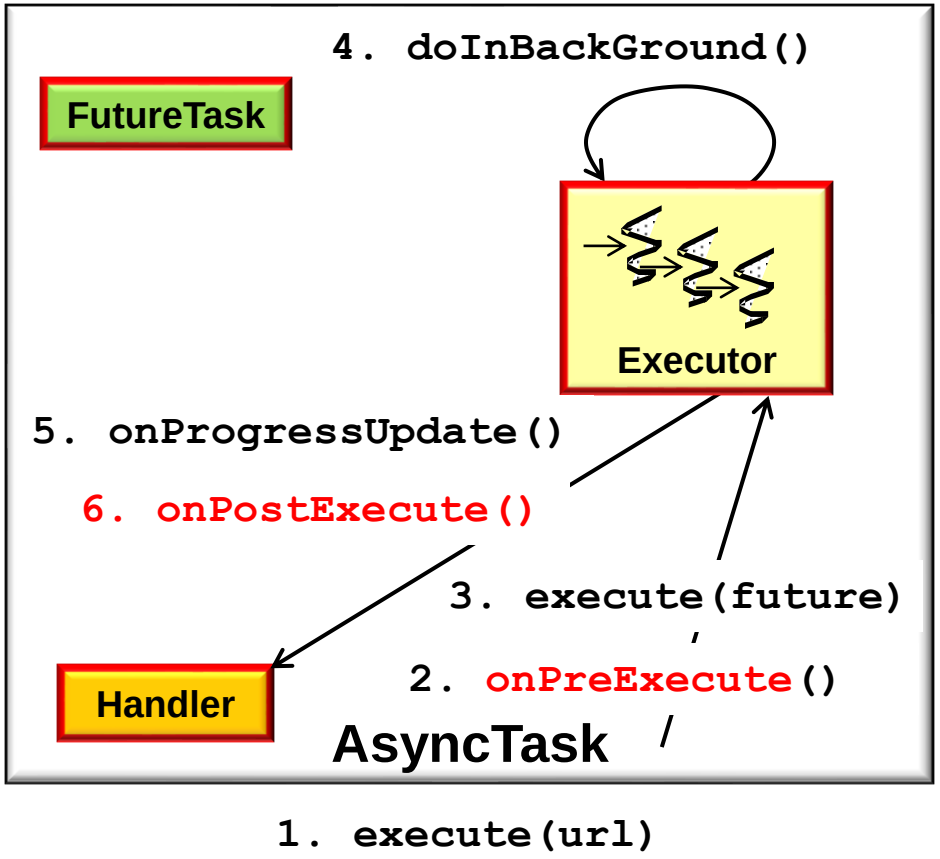





See en.wikipedia.org/wiki/Template_Method_pattern

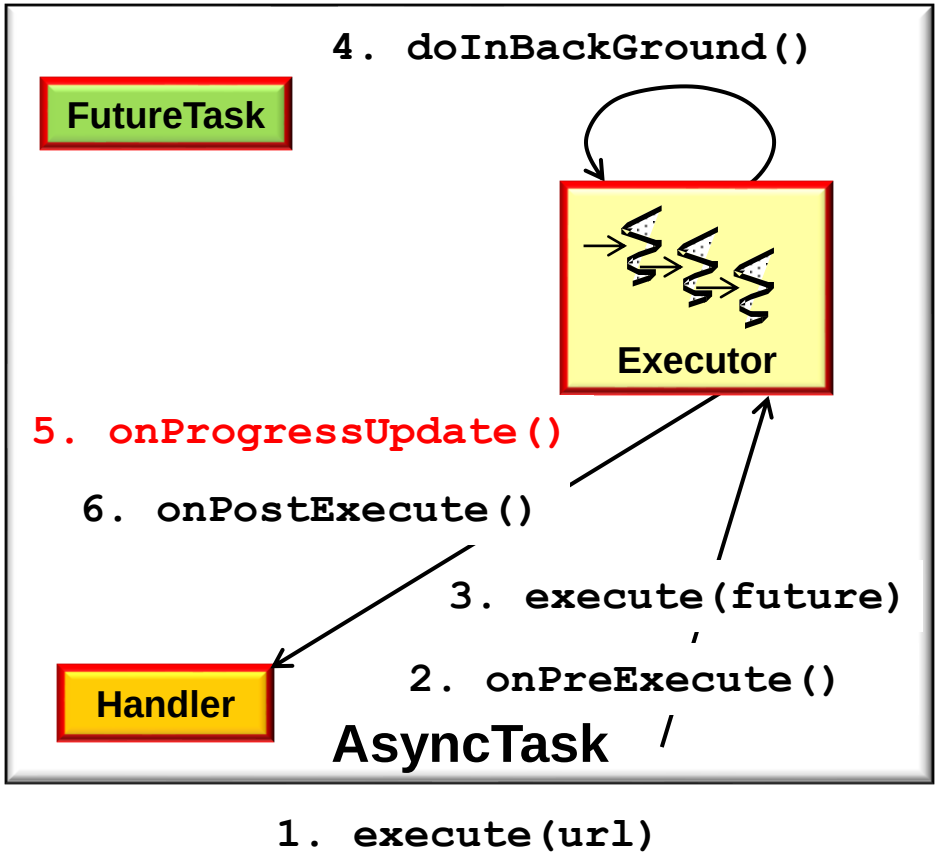
# Overview of the AsyncTask Framework

- In contrast, AsyncTask framework classes are more strongly connected
  - Complex framework details hidden via *Façade* pattern
  - Yields a smaller "surface area"
  - Tasks run concurrently, *without* manipulating threads, handlers, messages, or runnables directly
  - Hook methods can automatically
    - Perform pre-processing & post-processing in the UI thread



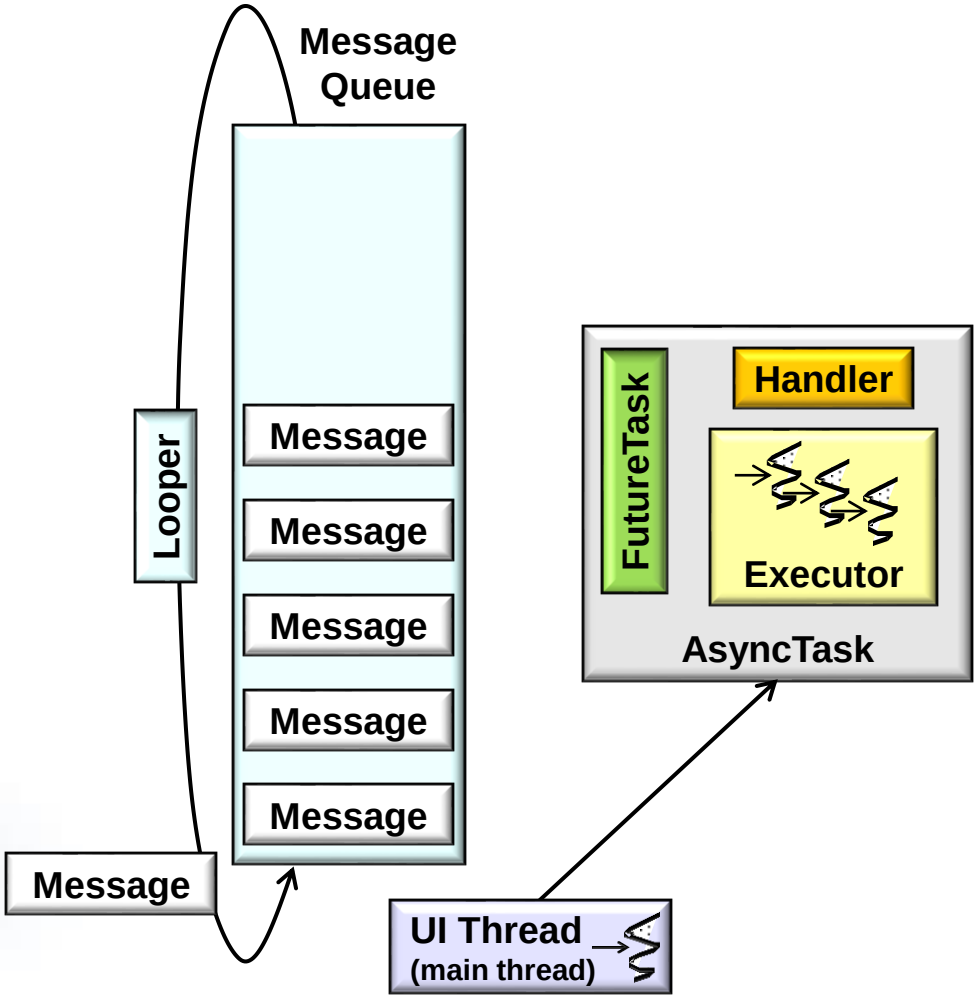
# Overview of the AsyncTask Framework

- In contrast, AsyncTask framework classes are more strongly connected
  - Complex framework details hidden via *Façade* pattern
  - Yields a smaller "surface area"
  - Tasks run concurrently, *without* manipulating threads, handlers, messages, or runnables directly
  - Hook methods can automatically
    - Perform pre-processing & post-processing in the UI thread
    - Pass typed date between background & UI threads



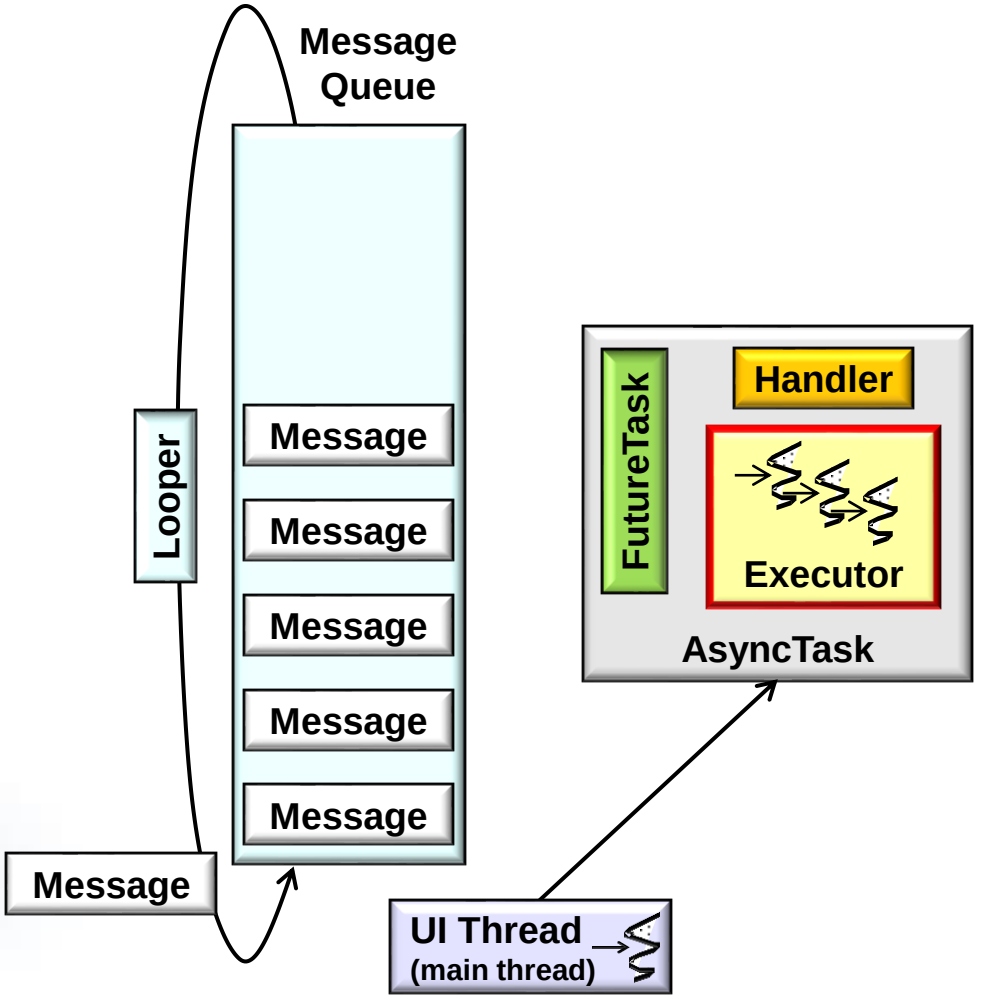
# Overview of the AsyncTask Framework

- Likewise, AsyncTask performance can be scaled up transparently



# Overview of the AsyncTask Framework

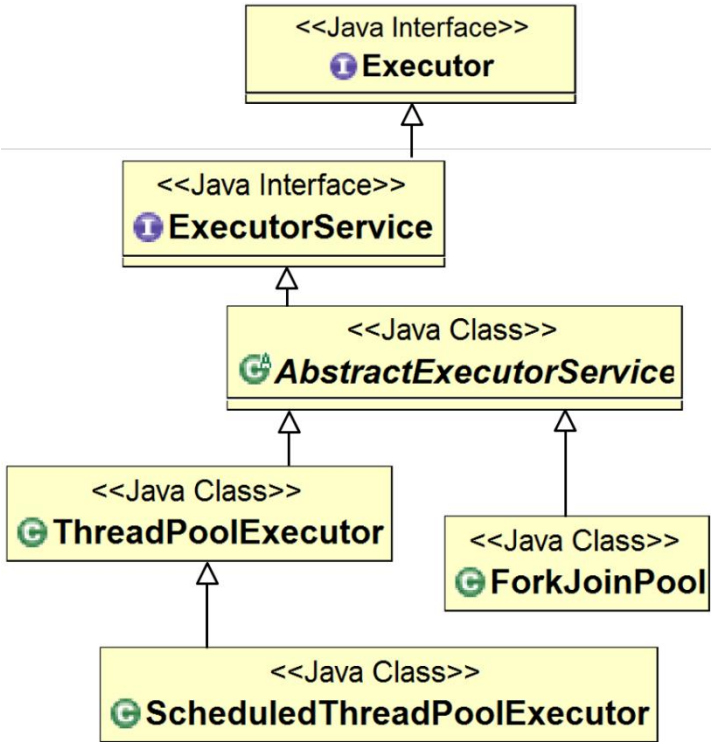
- Likewise, AsyncTask performance can be scaled up transparently, e.g.
  - Via a thread pool specified using policies or programmatically

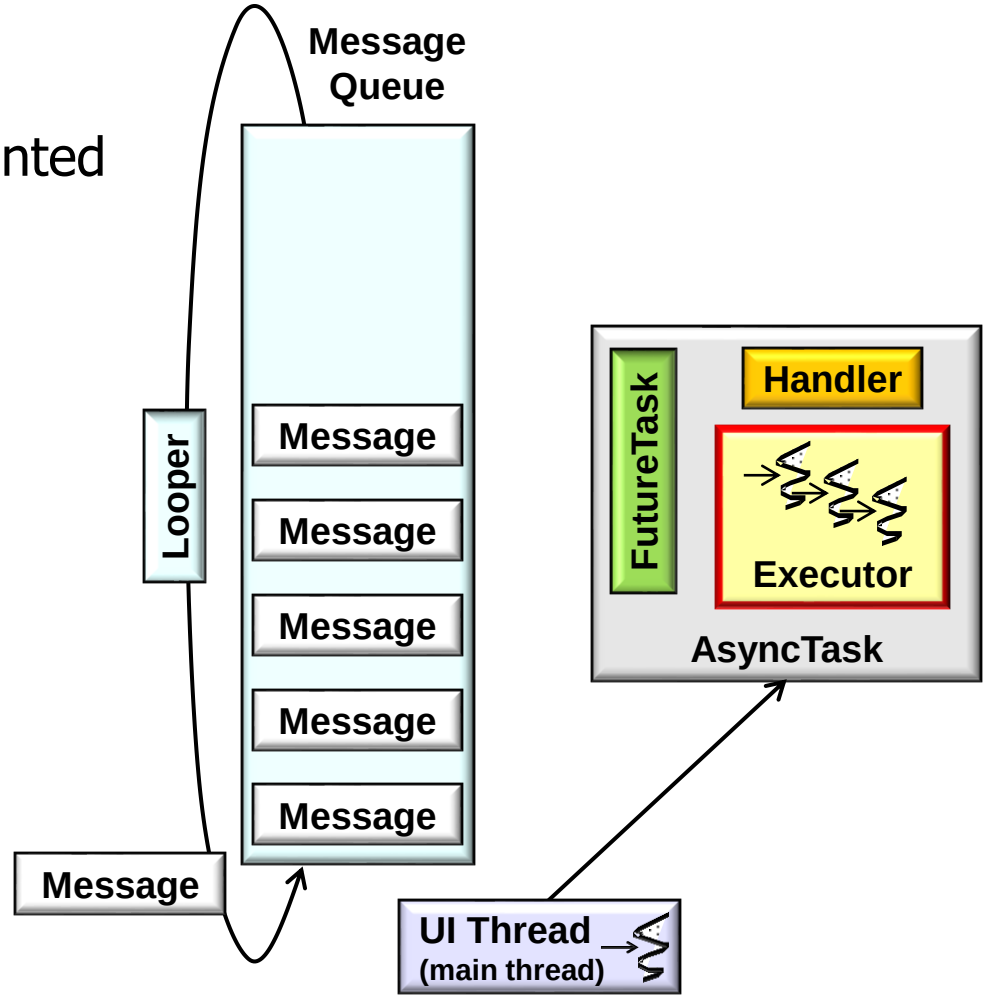



See en.wikipedia.org/wiki/Thread_pool

# Overview of the AsyncTask Framework

- Likewise, AsyncTask performance can be scaled up transparently, e.g.
  - Via a thread pool specified using policies or programmatically
  - This thread pool can be implemented via the Java Executor framework




See docs.oracle.com/javase/tutorial/essential/concurrency/executors.html

# End of the AsyncTask Framework: Introduction